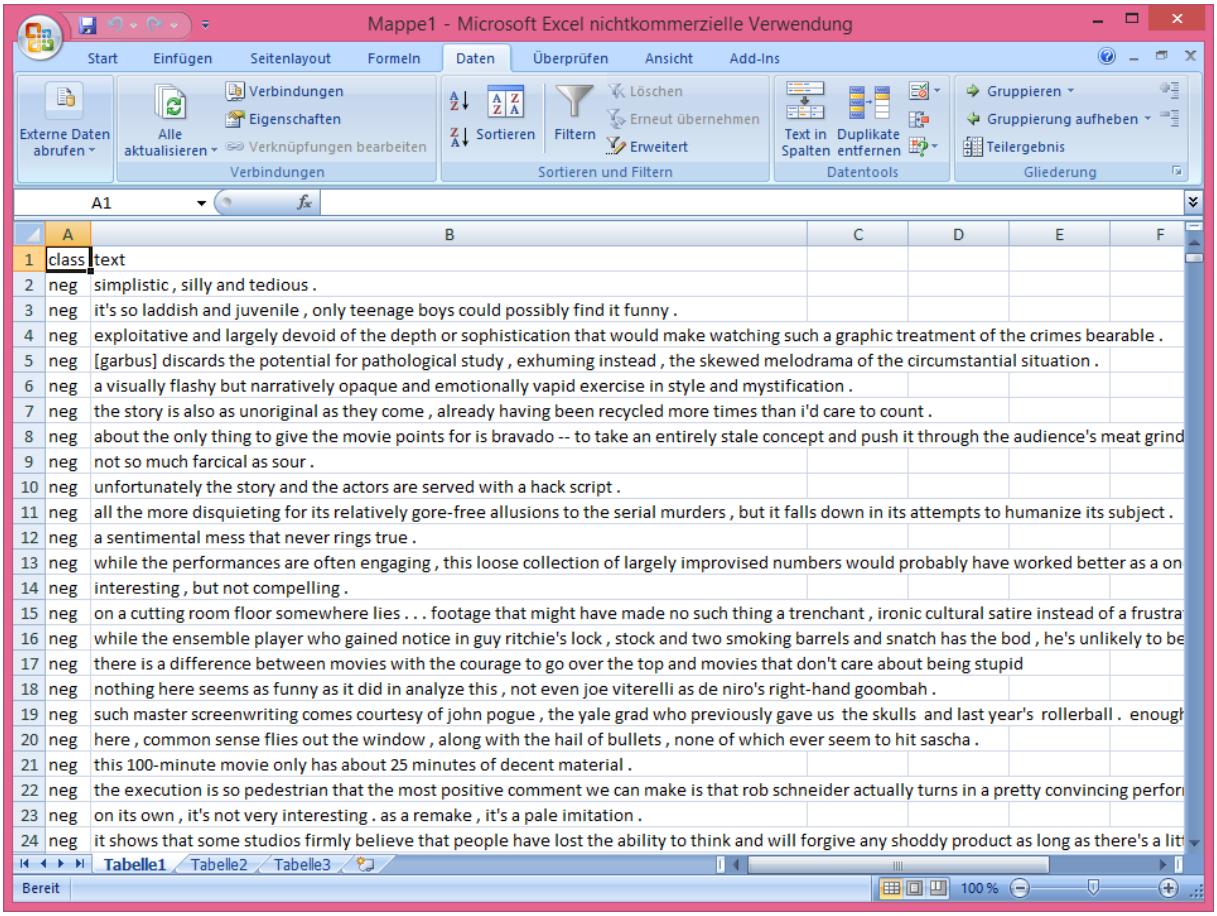
DALMOOC Week 8 – LightSide – Some Text Mining

14.12.14 / Ingrid Dethloff http://blog.idethloff.de

- **LightSide's Example Dataset "sentiment_sentences"**
- 10662 instances, two columns: class (neg/pos) and text



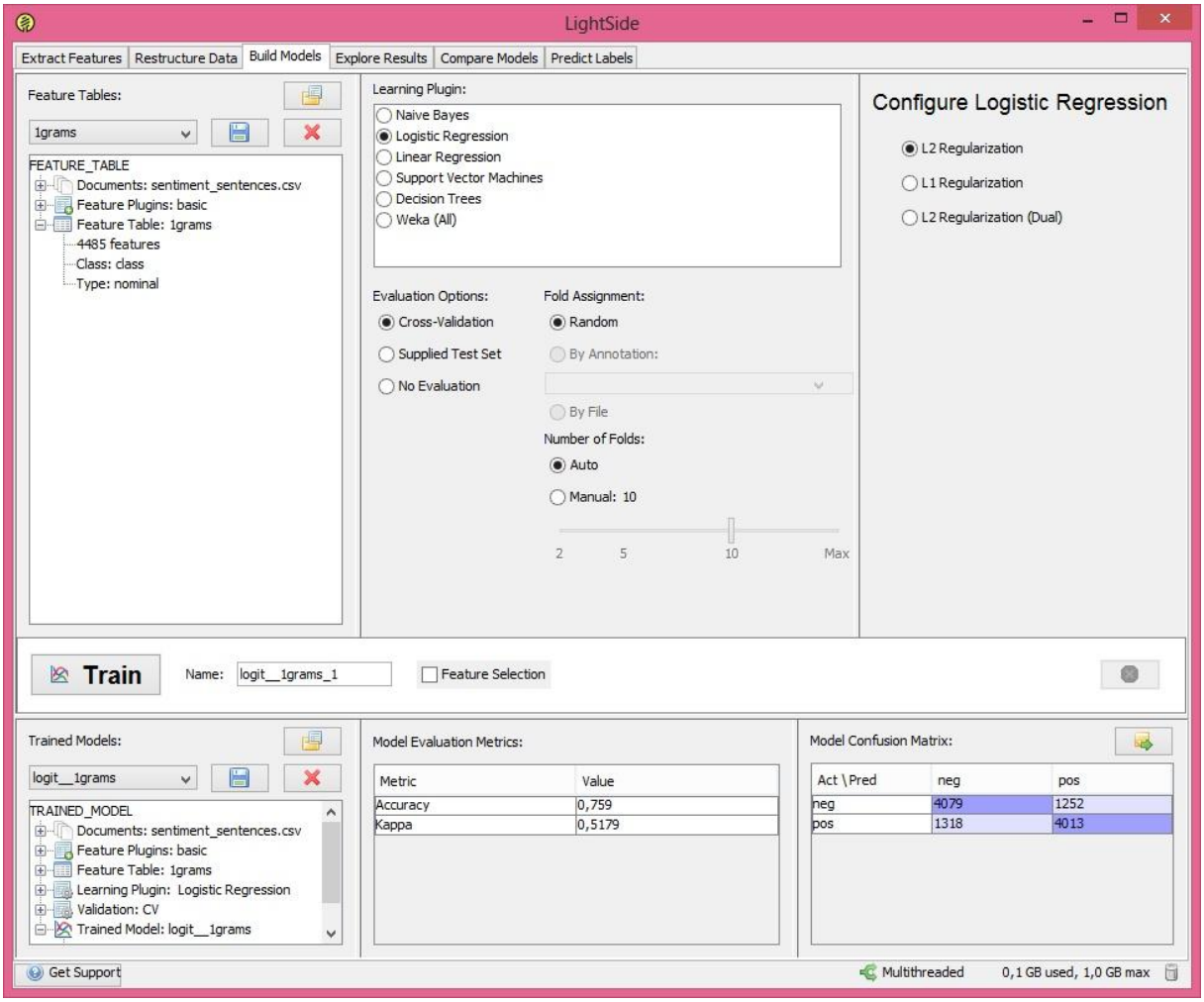
LightSide: Feature Extractor Plugins = Basic Features

01) Unigrams & Logistic Regression

Configure Basic Features = Unigrams & Include Punctuation & Track Feature Hit Location

Build Models = Logistic Regression & Cross Validation 10fold

➔ **Result: 4485 Features // Accuracy = 0,759 // Kappa = 0,518**

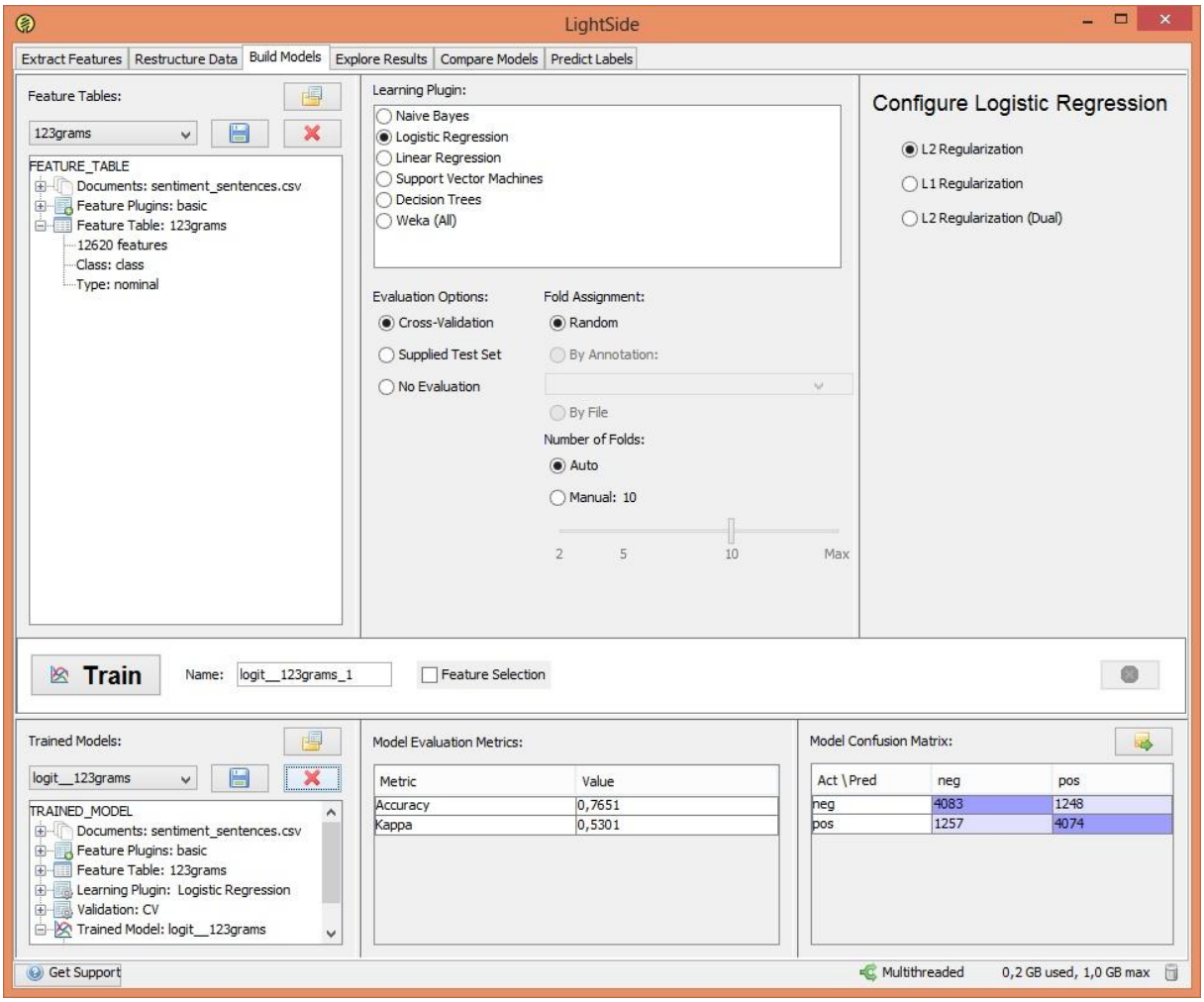


02) Unigram, Bigram, Trigram & Logistic Regression

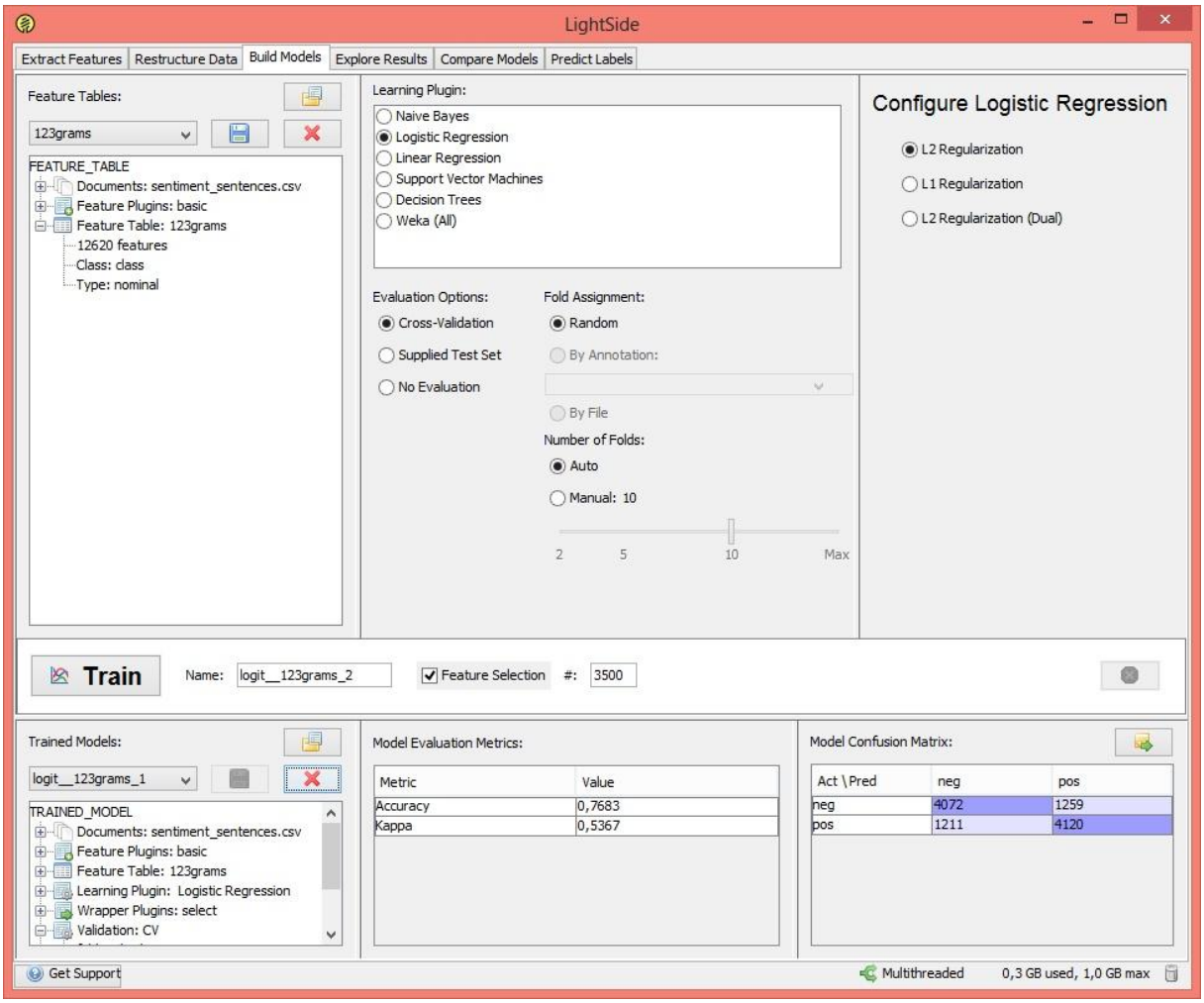
Configure Basic Features = Unigrams, Bigrams, Trigrams & Include Punctuation & Track Feature Hit Location

Build Models = Logistic Regression & Cross Validation 10fold

➔ **02a) Whole Feature space of 12620 Features // Accuracy = 0,765 // Kappa = 0,530**



➔ **02b) Feature Selection -> 3500 Features // Accuracy = 0,768 // Kappa = 0,537**



03) Model Comparison Unigrams / Unigrams, Bigrams, Trigrams

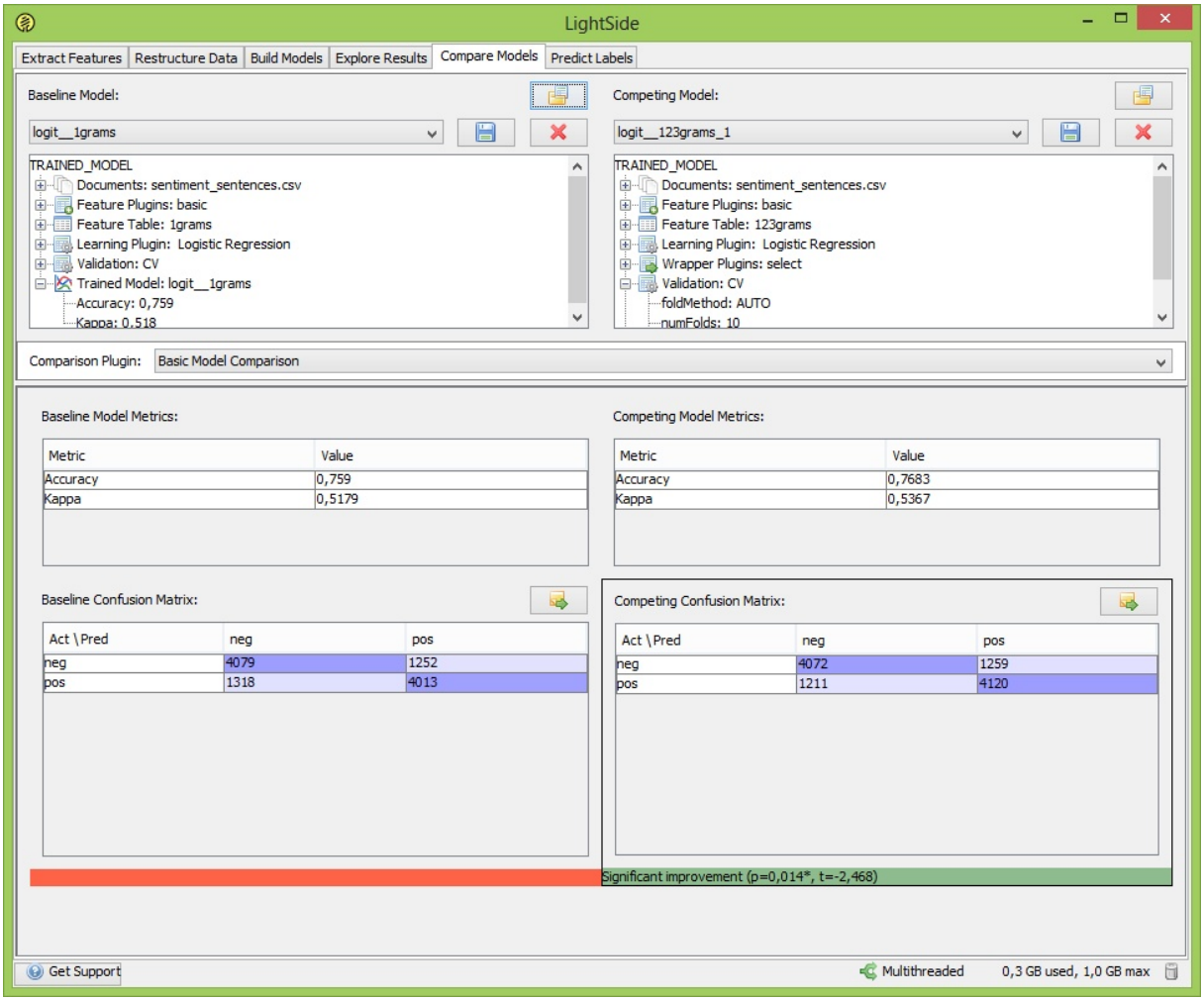
Number 01: Unigrams & Logistic Regression (All Features)

Number 02b: Unigrams, Bigrams, Trigrams & Logistic Regression & Feature Selection = 3500

➔ **Significant Improvement (p=0,014*, t=2,468)**

Model 01 predicted 1252 as positive and 1318 as negative when the data said the opposite

Model 02b predicted 1259 as positive and 1211 as negative when the data said the opposite

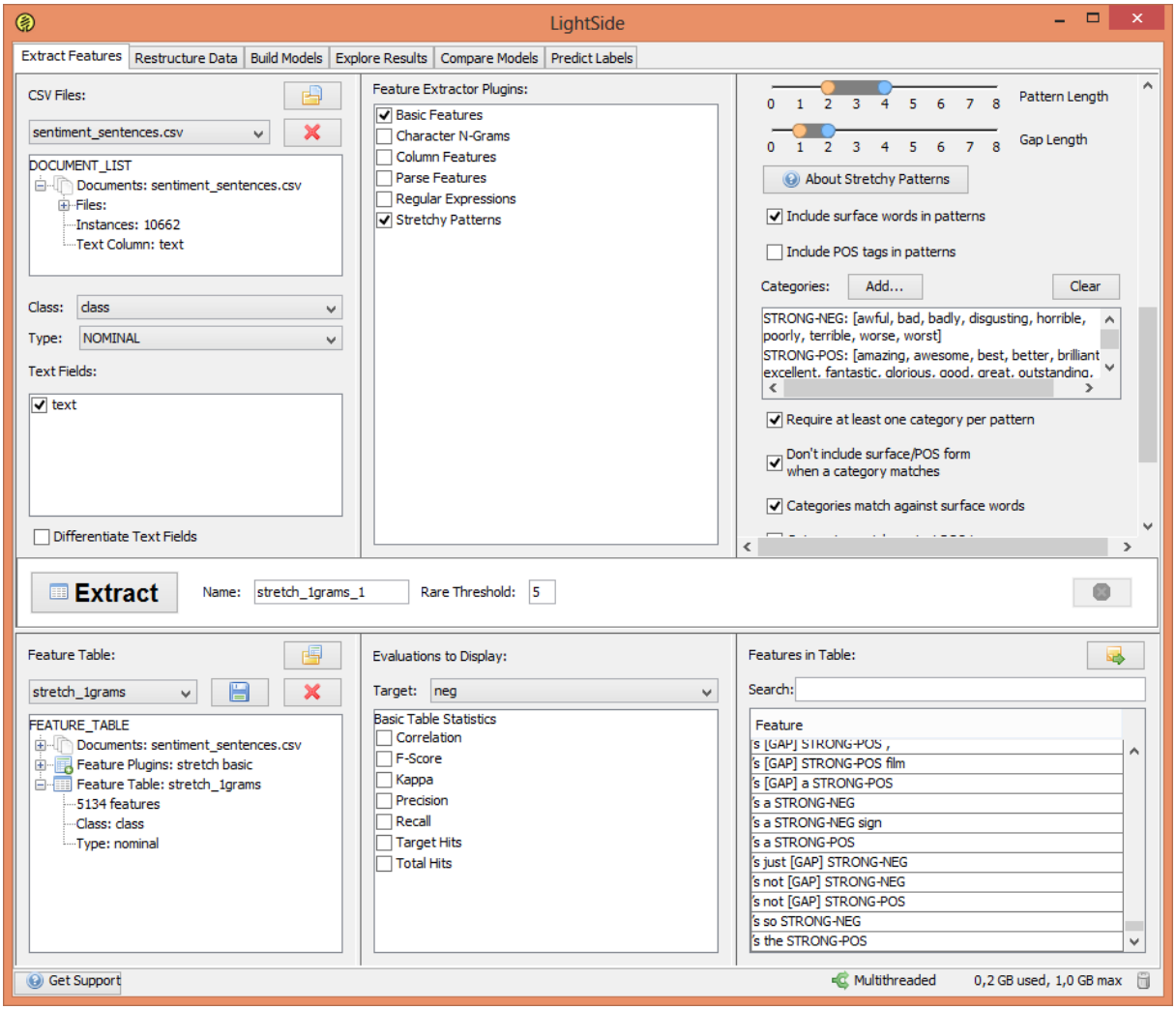


LightSide: Feature Extractor Plugins = Basic Features & Stretchy Patterns

04) Unigrams & Logistic Regression

01a) Configure Basic Features = Unigrams & Include Punctuation & Track Feature Hit Location Configure Stretchy Patterns = (default Pattern Length=2-4 / Gap Length = 1-2) // Add LightSide Categories negative.txt and positive.txt) // check off “Require at least one category per pattern”

-> Result: 5134 Features

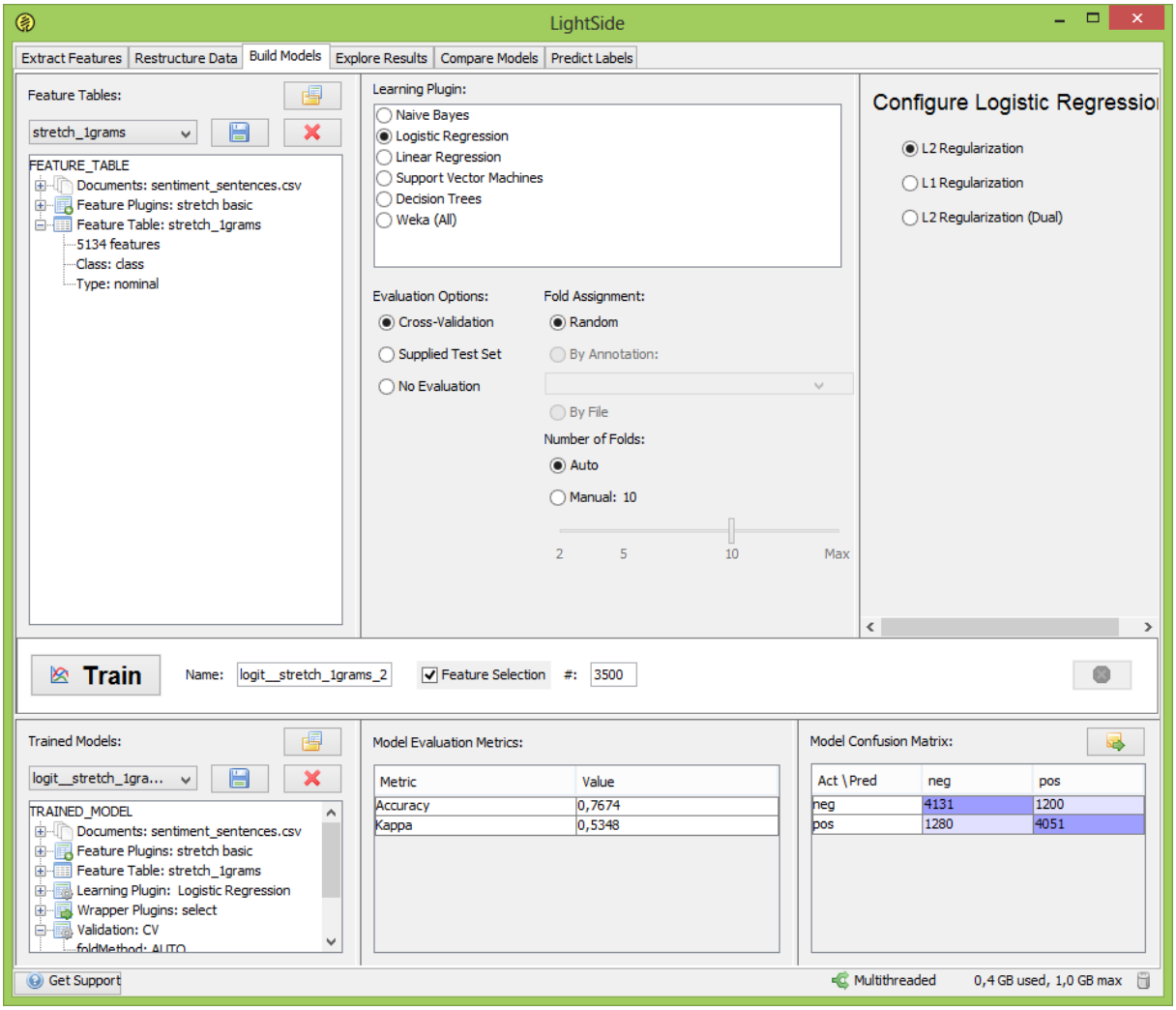


04b) Build Models = Logistic Regression & Cross Validation 10fold

➔ **Result: 5134 Features // Accuracy = 0,759 // Kappa = 0,517**

04c) Build Models = Logistic Regression & Cross Validation 10fold & **Feature Selection = 3500**

➔ **Result: 3500 Features // Accuracy = 0,767 // Kappa = 0,535**



05) Model Comparison Unigrams / Unigrams, Stretchy Patterns

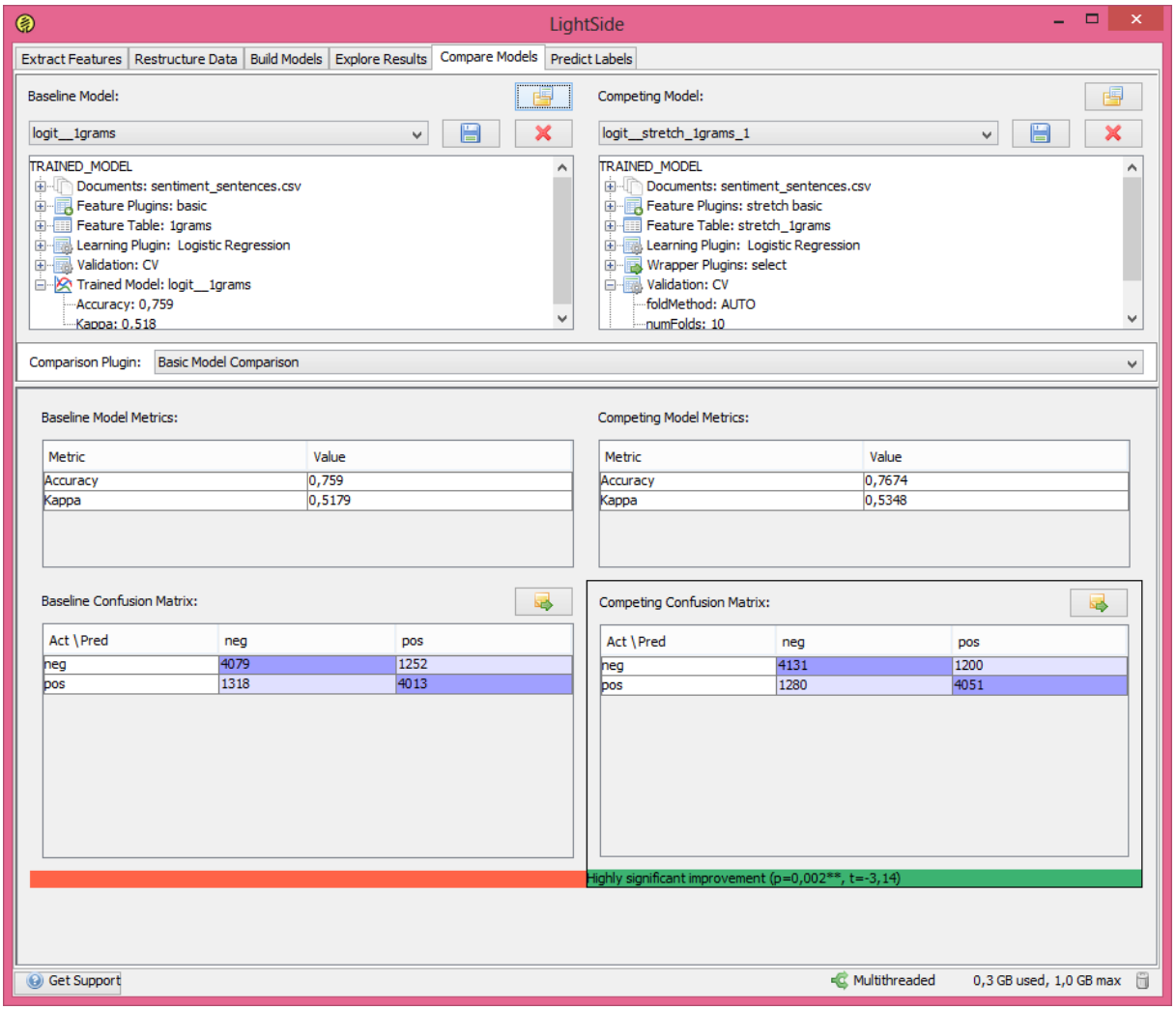
Number 01: Unigrams & Logistic Regression (All Features)

Number 04c: Unigrams & Stretchy Patterns & Logistic Regression & Feature Selection = 3500

➔ **Highly Significant Improvement (p=0,002*, t=3,14)**

Model 01 predicted 1252 as positive and 1318 as negative when the data said the opposite

Model 04c predicted only 1200 as positive and 1280 as negative when the data said the opposite

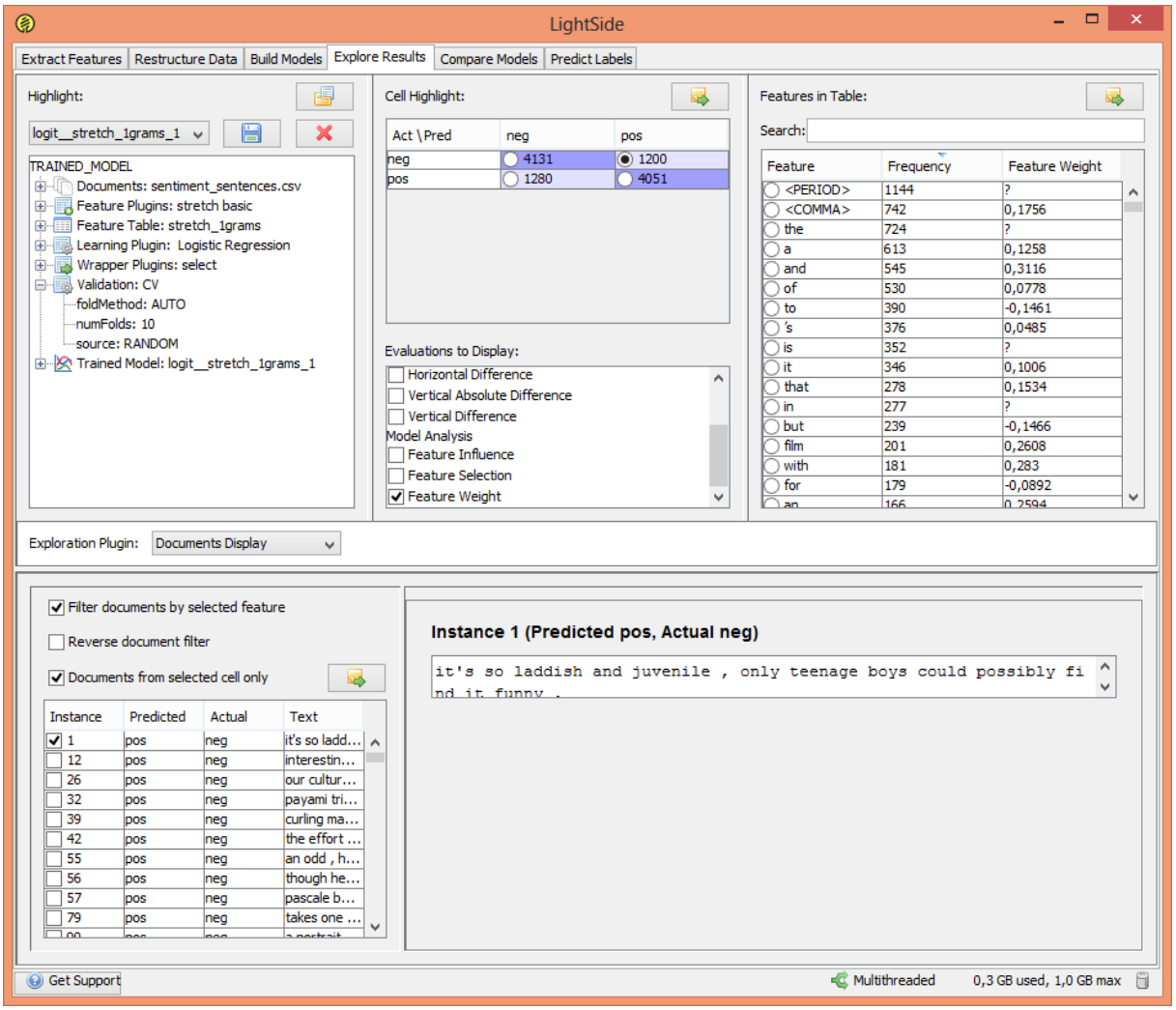


06) Explore Feature Space Model 04c: Unigrams & Stretchy Features & Feature selection=3500

Configure Confusion matrix select: Data negative & Prediction Positive = 1200

Configure “Evaluations to Display”: check off “Frequency” and “Feature Weight” // Sorting by Frequency

Configure “Exploration Plugin” = Documents Display // check off “Filter Documents by selected feature” and “Documents from selected cell only”



Task: Use this interface to explore which features got the most weight in your model. It’s most important to consider features that both got a lot of weight and occurred more than just a couple of times. Which features were most important? What did the stretchy pattern features add?

<u>Data negative & predicted positive</u>

- As I included punctuation in the Basic Features Extraction, these (period, comma) have the highest frequency and lowest or no weight at all. Other features with a high number for false positives (frequency /weight) are: “you” (=135 / 0,54 -> normally positive term), “n’t” (103 / -

0,476 -> normally negative term), “all” (67 / -0,333 -> normally negative term), “what” (58 / 0,375 -> normally positive term), “so” (50 / 0,361 -> normally positive term), “way” (40 / 0,148 -> normally positive term), “love” (37 / 0,441 -> normally positive term)
The feature “n’t” as a negative form would be associated with negative sentiment and when you check this feature in the confusion matrix for “data negative & prediction negative”, you get a high frequency of 456.

- Stretchy patterns added context, for example the feature “STRONG-POS [GAP] but” (9 times), which puts into perspective a positive term: In the original text, this would fit to “…good intentions, but”, “…great team, but”.
  The other way round, if you look at “data negative & predicted negative” in the confusion matrix, there are more features of this kind: “STRONG-POS [GAP] but”, “STRONG-POS [GAP] but the”, “STRONG-POS [GAP] , but”, “STRONG-POS [GAP] . but” etc. The feature “but” which stands alone then has a weight of 0,147 which is so small that it indicates (in my understanding), that from this word alone, you can’t predict if something is meant positive or negative.